UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/809,499	03/15/2001	Gerard J. Holzmann	Holzmann 17	5667

27964 7590 09/13/2004

HITT GAINES P.C.
P.O. BOX 832570
RICHARDSON, TX 75083

EXAMINER	
STEELMAN, MARY J	

ART UNIT	PAPER NUMBER
2122	

DATE MAILED: 09/13/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/809,499	HOLZMANN, GERARD J.	
	Examiner	Art Unit	
	Mary J. Steelman	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *17 May 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-23* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-23* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-23 are pending

Drawings

2. In view of the amendments to the Specification, the objections to Figures 2 & 3 are hereby withdrawn.

3. Regarding figure 1, Examiner does not have a copy of the drawing to review for corrections.

Specification

4. In view of the amendments to the Specification, the objections in the prior Office Action are hereby withdrawn.

Claim Objections

5. In view of the amendments to the claims, the objections in the prior Office Action are hereby withdrawn.

Claim Rejections - 35 USC § 112

6. In view of Applicant's comments and the amendments to the Specification the 35 USC 112 rejections are hereby withdrawn.

Requirement for Information – 37 CFR 1.105

7. Applicant and the assignee of this application are required under 37 CFR 1.105 to provide the following information that the examiner has determined is reasonably necessary to the examination of this application.

8. In response to this requirement, please provide: Holzmann et al., "Software Model Checking: Extracting Verification Models from Source Code", published in Proceedings of PSTV/FORTE99 (Kluwer, 1999), page 481.

9. In responding to those requirements that require copies of documents, where the document is a bound text or a single article over 50 pages, the requirement may be met by providing copies of those pages that provide the particular subject matter indicated in the requirement, or where such subject matter is not indicated, the subject matter found in applicant's disclosure.

10. The applicant is reminded that the reply to this requirement must be made with candor and good faith under 37 CFR 1.56. Where the applicant does not have or cannot readily obtain an item of required information, a statement that the item is unknown or cannot be readily obtained will be accepted as a complete reply to the requirement for that item.

11. This requirement is an attachment of the enclosed Office action. A complete reply to the enclosed Office action must include a complete reply to this requirement. The time period for reply to this requirement coincides with the time period for reply to the enclosed Office action.

Claim Rejections - 35 USC § 103

12. Claims 1- 23 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent 5,671,416 to Elson, in view of US Patent 6,330,530 to Horiguchi et al.

Elson disclosed modifying source code applicable to (col. 25, lines 33-40) electrical design and the design of complex integrated circuits and simulation of functions of VLSI devices prior to commitment to silicon (verification).

Per claim 1, Elson disclosed:

-generating a parse tree defining a control flow from the source code for procedural elements thereof;

(Elson: Col. 5, lines 13-16, "The compiler...receives source code..."compilation of the source code into a parse tree is performed...", col. 5, lines 25-26, "Parse tree data base is used in queries and for modifying the source code using query/rewrite processor...", col. 10, lines 37-41, "Each element (or node) of the tree corresponds to a symbol of the language definition (procedural elements), and that node's subtrees correspond to the symbols in the rule that the syntax analyzer as determined to be correct at that point...", col. 11, lines 23-41, "Query Search Process There are three major data structures: a tree (a parse tree, for example), a finite state automaton (FSA), and a stack of pointer to FSA states and tree nodes...There is one start state in the FSA..." There is a control flow of the parsed source code represented in a finite state machine, that is used in a query process to rewrite/translate the program source code (col. 2, line 63-col. 3, line11).)

-identifying source code elements;

(Elson: Col. 2, lines 56-57, "...it generates a parse tree with tokens (source code elements) and pointer tables interrelating the tokens...")

Elson failed to disclose specifics regarding "strings". However Horiguchi disclosed more details regarding transforming a source language linguistic structure (strings) into a target language. Horiguchi disclosed:

-from the parse tree, generating source strings for selected ones of the source code elements; (Horiguchi: See fig. 2B. Col. 7, lines 5-9, "Syntactic analysis module uses parsing grammar to create a syntax parse tree for the sentence. Parsing grammar contains the source language context-free grammar rules in the form of a parsing table and the associated rule bodies…")

-defining corresponding default conversions for translating the source strings into a target language of a model checker;

(Horiguchi: Fig. 2A, Fig. 3, #220 and fig. 7A. Col. 8, lines 64-65, "…transfer model searches bilingual example database for matches (default conversions)…", col. 5, line 56-col. 6, line 12, "All rule bodies utilized by the grammars of language translation system are in the form of computer-executable routines produced by defining the grammar in terms of a grammar programming language and passing appropriate rule bodies through a GPL compiler. The output may be in the form of directly executable code…" Also col. 8, lines 17-22, "…the resulting transfer generation tree is used by transfer module to match the feature structure against the example database." Also, col. 9, lines 28-34, "…Matching module searches thesaurus to fine the most specific thesaurus entry that dominates a thesaurus code from each representation (verification)." Col. 12, line 54, "…generate target language feature structure…" and col. 13, lines 10-14, "The process blocks of fig. 8 and fig. 9 would be applied to word or slot until a good match is found…recursively executed until a match of the entire SLS is processed.")

-generating a verification model in the target language wherein the verification model conforms to the control flow and to the corresponding default conversions for the selected ones of the source code elements.

(Elson: Col. 25, lines 13-17, "...the principle of generating a parse tree in combination with an appropriately modified query language make this invention applicable in many other areas. The present invention is applicable to any type of structure which can be e3xpanded into a tree...", col. 25, lines 33-40, This process is applicable to...electrical designs and the design of complex integrated circuits...", col. 25, lines 38-53, "Simulation of functions of VLSI devices (verification models) prior to commitment to silicon, is another application area where the present invention provides significant advantage..." The conforming control flow is ensured by the query parse tree that provides corresponding conversion of the source code elements. Col. 2, line 63-col. 3, line 11, "...compiler allows navigation through the parse tree...supports searching for a declared symbol or program term (source code elements)...allows to delete, modify, replace...The new language (target language)...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, (col. 1, lines 28-30), "prepare a parsed representation of the original source program..."through analyzing the parsed tokens, creating a target language, including Horguchi's string specifics, because (col. 1, lines 38-43), "The use of various types of data and corresponding pointers allows a user to implement changes based on the parsed representation of the program and to reconstruct the original source

code...making use of the query language of the present invention." Both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 2:

-optionally searching a conversion table for an entry associated with at least one of the source strings, the entry including a translation for the at least one of the source strings; (Horiguchi: Claim 7, lines 50-52, "Transfer module uses GPL rule bodies within transfer grammar to match the input source sub-structures of slots to the source...in example database...")

-substituting the translation for the corresponding default conversion for the at least one of the source strings, wherein the verification model further conforms to the translation. (Horiguchi: Col. 8, lines 7-8, "If the application of each rule succeeds, a child rule-node...is added to tree. If the application fails, the s-node is tagged as 'dead' for subsequent removal...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 3:

-source code elements include basic statements and Boolean conditionals.

(Horiguchi: Col. 9, lines 1-5, "…for example, full sentences, ("How do you do?", "May I help you?") (Boolean), verb-phrases ("I have an appointment.", "I have dinner") (basic statements) , …)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 4:

-generating of source text strings includes the further step of expressing the source text strings in a canonical form.

(Horiguchi: Col. 3, lines 39-46, "The grammar rules are recursively applied to the SLS sub-structures from a top-most transfer rule until all SLS sub-structures (source) within the SLS are transferred to corresponding TEF sub-structures (target)…" Grammar rules are used to omit redundant white space or insert / delete certain characters, providing for unambiguous precedence (canonical form).)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 5:

-specifics of the corresponding default conversions can depend on a usage of the selected ones of the source code elements.

(Horiguchi: Col. 6, lines 50-52, "The feature structures built by morphological analysis module are input into lexical ambiguity reduction module…" Col. 12, lines 34-39, "If an exact match is not found, thesaurus matching system is used to define how good a match has been found…" Col. 13, lines 44-46, "Matching module searches thesaurus to fine the most specific thesaurus entry that dominates a thesaurus code from each representation. Matching module retrieves previously calculated relative entropy values from slots…")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 6:

-conversion table further includes samples of source strings.

(Horiguchi: Col. 8, lines 64-65, "…transfer module searches bilingual example database (samples of source strings) for matches…")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 7:

-conversion table further includes classes of source strings.

(Horiguchi: col. 6, lines 16-19, "A morphological analysis module takes text input and uses a source language dictionary to decompose the words into morphemes by identifying root forms, grammatical categories, and other lexical features of the words.", and col. 11, lines 1-7, "...example database may change as the context of the language changes. That is, example database changes as the context changes between travel language, medical language, legal language...(classes of source strings)")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 8:

-searching of the conversion table includes the step of pattern matching the at least one of the source strings to the samples of source strings.

(Horiguchi: Col. 6, lines 16-19, "A morphological analysis module takes text input and uses a source language dictionary to decompose (for pattern matching)...", col. 6, lines 64-67, "Lexical ambiguity reduction module weighs the cost assigned to each word in the sentence and selects (pattern matching) those feature structures that have the lowest cost.")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 9:

-searching of the conversion table includes the step of pattern matching the at least one of the source strings to the classes of source strings.

(Horiguchi: Col. 11, lines 1-7, "...example database may change as the context of the language changes...travel language, medical language, legal language...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 10:

-corresponding default conversions causes the translating of the source strings to respective equivalent statements in the target language when the selected ones of the source code elements are fully relevant to a property to be tested, the translating of the source strings to null statements in the target language when the selected ones of the source code elements are irrelevant to the property to be tested, and the translating of the source strings to preservation statements in the target language when the selected ones of the source code elements are partially relevant to the

property to be tested, preservation statements being statements that preserve a relevant part of the source strings and that suppress an irrelevant part of the source strings.

(Horiguchi: Col. 8, lines 5-22, "Transfer grammar rules added to tree are applied to the s-nodes. If the application of each rule succeeds (use default conversion)...If the application fails (irrelevant) ...The process is repeated until all sub-features (break down into sub-features when partially relevant) in the target language associated with a match are substituted for the corresponding ...in the source language." and col. 10, lines 1-4, "When no match is found, a number of back-up rules within grammar are applied...(use variety of rules if exact match is not found)")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 11:

-generating a verification model step includes the further step of translating ones of the source strings to a non-deterministic choice of possible outcomes.

(Horiguchi: Col. 5, line 65-col. 6, line 5, "All rule bodies utilized by the grammars of language translation system are in the form of computer-executable routines produced by defining the grammar in terms of a grammar programming language (GPL) and passing appropriate rule bodies through a GPL compiler (verification model). The output of the GPL compiler may be in the form of directly executable code..." Also, col. 6, lines 57-67, "Each possible combination

(choice of possible outcomes) of adjacent segmented words are assigned a lexical cost. Dictionary defines combinations of words...Lexical ambiguity reduction module evaluates each feature structures that contains a part-of-speech ambiguity...selects those feature structures that have the lowest cost.")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 12:

-generating a verification model step includes the step of populating the control flow with the translated source strings.

(Horiguchi: Col. 8, lines 19-22, "The feature structures and sub-structures in the target language associated with a match are substituted...", and col. 8, lines 49-56, "The leaf nodes contain output feature structures that represent valid sentences when the syntactical generation tree is complete. The sequence of output feature structures that represents the best sentence is converted into output text by using the dictionary, and the thesaurus...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 13:

-default conversion includes a keep, the keep causing the generating of a verification model step to provide an equivalent statement in the target language.

(Horiguchi: Col. 8, lines 7, "If the application of each rule succeeds, a child rule-node is added to tree…")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 14:

-default conversion comprises a hide, the hide causing the generating of a verification model step to provide a null statement in the target language.

(Horiguchi: Col. 8, lines 8-10, "If the application fails, the s-node is tagged as 'dead' for sub-sequent removal.")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 15:

- default conversion comprises a print, the print causing the generating of a verification model step to embed the respective source strings in a print statement in the target language."
(Horiguchi: Col. 10, lines 1-5, "When no match is found, a number of back-up rules within grammar are applied to allow a simple, rule-based treatment of the unmatched slot or syntactic structure. For example, to allow a direct transfer (embed / print the source string in the target) or to delete the structure.")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claims 16 and 17:
-simplifying step includes the steps of: removing nodes corresponding to null statements; removing nodes successive to false nodes; skipping selected nodes mapped to true.
(Horiguchi: Col. 8, lines 7-10 and 14-19, "If the application of each rule succeeds...If the application fails...Transfer generation tree is then pruned (simplifying, remove nodes corresponding to null statements) to remove any 'dead' nodes and corresponding sub-trees. If root is tagged as 'dead,' the generation fails. Otherwise, the resulting transfer generation tree is used by transfer module to match the feature structure...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string

transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 18:

- collecting certain data object information for nodes in the parse tree corresponding to basic statements in the source code, the certain data object information including definition information and use information;

(Horiguchi: Col. 6, lines 21-25, "Feature structures are well known in the art as linguistic data structures that contain feature-value pairs (definition information and use information) for strings, symbols, and numbers that appear in a natural language sentence. Each feature of a word is mapped to the appropriate value...", col. 7, lines 9-18, "Each leaf of the syntax parse tree is a feature structure for one of the words in the sentence. Once the leaves are created, an intermediate feature structure for each branch node in the syntax parse tree is built...The rule body for each potentially applicable context-free rule could create a valid phrase from the possible combinations...")

-constructing a data dependency graph for the source code based upon the collected data object information, the data dependency graph having data dependency graph nodes corresponding to a data object, the data dependency graph having directed edges from first data dependency graph nodes to successive data dependency graph nodes if the successive data dependency graph nodes are used at least, once in a definition of the first data dependency graph nodes; (Col. 10, lines 9-

13, "Source language example feature structure may include a number of source language example slots. Each slot may in turn contain a number of nested slots (dependencies) …")

-determining a transitive closure for the data dependency graph dependency relation; (Col. 7, line 10, "Each leaf (last node of a branch) of the syntax parse tree is a feature structure for one of the words in the sentence.")

-adding edges to the data dependency graph according to the transitive closure, the adding step providing a second data dependency graph; (See nested sub-structures (second dependency graph), col. 10, lines 9-13.)

-for nodes corresponding to basic statements in the source code having translations other than hide or print, marking second data dependency graph data objects with identifiers corresponding to the definition information and the use information; (Col. 6, line 21, "feature-value pairs")

-for nodes corresponding to basic statements in the source code having a hide translation; marking second data dependency graph data objects with a hide identifier; checking the second data dependency graph, data objects for identifiers and the hide identifier. (Col. 8, lines 8-10, "If the application of each rule succeeds, a child rule-node is added to tree…if the application fails, the s-node is tagged as 'dead' for subsequent removal." And lines 15-16, "…pruned to remove any 'dead' nodes and corresponding sub-trees (second data dependency graph data objects

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, by including Horguchi's string transformation specifics, because both inventions are related to transforming a source language. Horiguchi merely provides more details on transforming strings.

Per claim 19:

-A method for verifying that a software based system satisfies certain properties...

(Elson: Abstract, line 1, "apparatus..."(system)).

-extracting a finite state model from the source code, the extracting step including the steps for:

(Elson: Col. 11, lines 27, "....finite state automaton...", col. 11, lines 3-34, "The stack of pointers to FSA states and tree nodes...")

-abstracting the source code statements based upon relevancies between the certain properties and the source code statements;

(Elson: Col. 13, lines 29-33, Once a parse tree is established the new parse tree query language...can be used in a first step to search for and locate (based upon relevancies) a selected term in the parse tree, and then, in a second step, the located term may be modified (abstracted) as needed using output instructions...")

-expressing the finite state model in an input language for a model checker;

(Elson: Col. 11, line 27, "...finite state automaton...", See FIG. 11, col. 13, lines 55-56, "FIG. 11 is an illustration of an arrangement of memory stacks defining a state diagram..."

-checking the finite state model for the certain properties in the code checker.

(Elson: Col. 11, line 27, "…finite state automaton…", See FIG. 11, col. 13, lines 55-56, "FIG. 11 is an illustration of an arrangement of memory stacks defining a state diagram during a search using the following search definition…", col. 14, lines 8-9, "The state diagram of a search defines at any time what search items have been detected…" As an example, col. 14, lines 41-42, "…nested loops (certain properties) can be accommodated by use of a state stack, which stores the occurrence of nested loops…")

Per clam 20:

-A system for verifying that a system satisfies certain properties…a model extractor operable to extract a finite state model from the source code, the model extractor implementing default conversions for translating selected source code elements and including:
(Elson: Col. 25, lines 38-40, "Simulation of functions of VLSI devices (model checker) prior to commitment to silicon, is another application area where the present invention provides significant advantage.", FIG. 11, (finite state machine extracted from the source code), col. 1, lines 62-64, "…a process which simplifies changing terms in a program written in a high level language…(translating selected source code)"

-a table of translations for translating other selected source code elements based upon defined abstractions;
(Elson: Col. 25, lines 44-45, "There is a need for accessing stimuli signals as well as behavior tales …to modify…")

-a translator responsive to the translations of the selected source code elements and the other selected source code elements for expressing the finite state model in an input language for a model checker;

(Elson: Col. 25, lines 38-46, col. 11, line 27-41, states in the FSA (finite state model).

-a model checker responsive to the certain properties and the finite state model for checking the finite state model for the certain properties.

(Elson: Col. 11, line 27, "...finite state automaton...", See FIG. 11, col. 13, lines 55-56, "FIG. 11 is an illustration of an arrangement of memory stacks defining a state diagram during a search using the following search definition...", col. 14, lines 8-9, "The state diagram of a search defines at any time what search items have been detected..." As an example, col. 14, lines 41-42, "...nested loops (certain properties) can be accommodated by use of a state stack, which stores the occurrence of nested loops...")

Per claim 21:

Elson disclosed:

-the model extractor further includes a parser for constructing a parse tree from the source code;

(Elson: Col. 2, lines 56-57, "...it generates a parse tree with tokens and pointer tables...")

wherein the translator translates source strings generated form the parse tree.

(Elson: Col. 17, lines 14-31, "...small program and a query...The example is demonstrated with the language, the program, its parse tree and the query operation...First, a few words regarding

the language and its definition...The remaining rules are syntactic expressions, with character string literals ued to denote operators and keywords..." Translator translates source strings as a part of the language definition.)

Per claim 22:

-the model extractor further operates to provide a control flow from the parse tree and to populate the control flow with translated source strings.

(Elson: Abstract, line 1-10, "...searching through a parse tree of a source code...uses a new type of parse tree in combination with a new query language and...processor...search through a parse tree fro the occurrences of one or more selected terms...and for easy generation of new source code differing from the original source code..." Parse tree sets up a language FSA, that is used to transform a source code. This is suitably used as a model extractor (col. 25, lines 38-53). The FSA provides state information (control flow). Translated source strings are a part of the language processed (col. 17, line 31).)

Per claim 23:

Elson disclosed

-A method for extracting a verification model from source code having a control flow for procedural elements of the source code...;

(Elson: Col. 2, line 45, "...method...", col. 5, lines 13-16, "The compiler...receives source code..."compilation of the source code into a parse tree is performed...", col. 5, lines 25-26, "Parse tree data base is used in queries and for modifying the source code using query/rewrite

processor...", col. 10, lines 37-41, "Each element (or node) of the tree corresponds to a symbol of the language definition, and that node's subtrees correspond to the symbols in the rule that the syntax analyzer (control flow for procedural elements of the source code) as determined to be correct at that point...", col. 11, lines 23-41, "Query Search Process There are three major data structures: a tree (a parse tree, for example), a finite state automaton (FSA), and a stack of pointer to FSA states and tree nodes...There is one start state in the FSA..." There is a control flow of the parsed source code represented in a finite state machine, that is used in a query process to rewrite/translate the program source code (col. 2, line 63-col. 3, line11). (Elson: Col. 25, lines 13-17, "...the principle of generating a parse tree in combination with an appropriately modified query language make this invention applicable in many other areas. The present invention is applicable to any type of structure which can be expanded into a tree...", col. 25, lines 33-40, This process is applicable to...electrical designs and the design of complex integrated circuits...", col. 25, lines 38-53, "Simulation of functions of VLSI devices (extracting a verification models from source code) prior to commitment to silicon, is another application area where the present invention provides significant advantage..." The conforming control flow is ensured by the query parse tree, FSA, that provides corresponding conversion of the source code elements. Col. 2, line 63-col. 3, line 11, "...compiler allows navigation through the parse tree...supports searching for a declared symbol or program term (source code elements)...allows to delete, modify, replace...The new language (target language)...")

Elson failed to disclose specifics regarding string translations. However Horiguchi disclosed:

-generating selected source strings from the source code;

(Horiguchi: See fig. 2B. Col. 7, lines 5-7, "Syntactic analysis module uses parsing grammar to create a syntax parse tree for the sentence.")

-translating ones of the selected source strings to corresponding target language statements according to default conversions;

(Horiguchi: Fig. 3, #220 and fig. 7A. Col. 8, lines 64-65, "...transfer model searches bilingual example database for matches (default conversions)..." Col. 12, line 54, "...generate target language feature structure..." and col. 13, lines 10-14, "The process blocks of fig. 8 and fig. 9 would be applied to word or slot until a good match is found...recursively executed until a match of the entire SLS is processed.")

-optionally searching a conversion table for entries associated with the selected source strings, the conversion table including a plurality of translations associated with various ones of the source strings;

(Horiguchi: Col. 9, lines 28-34, "Transfer grammar executes thesaurus matching system...Matching module searches thesaurus to find the most specific thesaurus entry that dominates a thesaurus code from each representation (plurality of translations associated with ones of the source strings)...")

-translating other ones of the selected source strings to corresponding target language statements according to the entries;

(Horiguchi: Col. 9, lines 28-34, "...Matching module searches thesaurus to fine the most specific thesaurus entry that dominates a thesaurus code from each representation (verification)." Also, col. 13, lines 10-14, "The process blocks of fig. 8 and fig. 9 would be applied to word or slot until a good match is found...recursively executed until a match of the entire SLS is processed. (other ones)")

-populating the control flow with the target language statements.
(Horiguchi: Col. 8, lines 19-22, "The feature structures and sub-structures in the target language associated with a match are substituted (populating)...", and col. 8, lines 49-56, "The leaf nodes contain output feature structures that represent valid sentences when the syntactical generation tree is complete. The sequence of output feature structures that represents the best sentence is converted into output (target language statements) text by using the dictionary, and the thesaurus...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify Elson's invention to modify source code, (col. 1, lines 28-30), "prepare a parsed representation of the original source program..."through analyzing the parsed tokens, creating a target language, including Horguchi's string specifics, because (col. 1, lines 38-43), "The use of various types of data and corresponding pointers allows a user to implement changes based on the parsed representation of the program and to reconstruct the original source code...making use of the query language of the present invention." Both inventions are related

to transforming a source language. Horiguchi merely provides more details on transforming strings.

Response to Arguments

13. Applicant's arguments filed 17 May 2004 have been fully considered but they are not persuasive.

Applicant has argued, in substance, the following:

(A) As noted on page 13, 4th paragraph, of Amendment A, received 17 May 2004, in reference to claim 1, "Horiguchi does not teach a method of extracting a verification model from program source code including generating a parse tree defining a control flow from the source code for procedural elements thereof."

Examiner's Response:

A new reference Elson, US Patent 5671416 has been provided to reject these limitations.

(B) As noted on page 14, 1st paragraph, Horiguchi does not address extracting from source code but instead translates from an input source language (not source code) to a target language."

Examiner's Response: An input source language can broadly be construed to meet the limitations of input source code.

Examiner maintains the rejections of claims 1-23.

Conclusion

14. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

15. This Office action has an attached requirement for information under 37 CFR 1.105. A complete reply to this Office action must include a complete reply to the attached requirement for information. The time period for reply to the attached requirement coincides with the time period for reply to this Office action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (703) 305-4564. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman 08/16/2004

ANTONY NGUYEN-BA
PRIMARY EXAMINER